TASK: UR20
CDRL: 00990

2

UR20 – Process/Environment Integration Ada Command Environment (ACE) Version 8.0 SunOS Implementation

Version Description Document

UNISYS

STARS-RC-00990/001/00

25 October 1990

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE 25 October 1990	3. REPORT TYPE AND DATES COVERED Version Description Document

4. TITLE AND SUBTITLE
Ada Command Environment (ACE)

5. FUNDING NUMBERS
STARS Contract
F19628-88-D-0031

6. AUTHOR(S)
William P. Loftus

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Unisys Corporation
12010 Sunrise Valley Drive
Reston, VA 22091

8. PERFORMING ORGANIZATION REPORT NUMBER
GR-7670-1163(NP)

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Department of the Air Force
Headquarters, Electronic Systems Division (AFSC)
Hanscom AFB, MA 01731-5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
00990

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release;
distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The Ada Command Environment (ACE) is an interactive Ada environment coupled with a set of Ada abstract data types (ADTs). The interactive environment allows users to rapidly prototype general Ada applications, while the ADTs allow prototyping of applications for particular domains, such as X Window System applications. In addition, the ADTs provide an Ada view of underlying applications, which when combined with the interactive environment replaces the traditional role of a command language. When using ACE, Ada becomes the command language as well as the programming language. This version of ACE includes support for X Window System prototyping.

[illegible]

14. SUBJECT TERMS
Ada Command Environment (ACE)
Abstract Data Types (ADT)

[illegible]

15. NUMBER OF PAGES
42

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT SAR

TASK: UR20
CDRL: 00990
25 October 1990

VERSION DESCRIPTION DOCUMENT
For The
SOFTWARE TECHNOLOGY FOR ADAPTABLE, RELIABLE SYSTEMS (STARS)

Ada Command Environment (ACE)
Version 8.0
SunOS Implementation

STARS-RC-00990/001/00
Publication No. GR-7670-1163(NP)
25 October 1990

Data Type: A005, Informal Technical Data

CONTRACT NO. F19628-88-D-0031
Delivery Order 0002

Prepared for:

Electronic Systems Division
Air Force Systems Command, USAF
Hanscom AFB, MA 01731-5000

Prepared by:

Unisys Defense Systems
Tactical Systems Division
12010 Sunrise Valley Drive
Reston, VA 22091

TASK: UR20
CDRL: 00990
25 October 1990

VERSION DESCRIPTION DOCUMENT
For The
SOFTWARE TECHNOLOGY FOR ADAPTABLE, RELIABLE SYSTEMS (STARS)

Ada Command Environment (ACE)
Version 8.0
SunOS Implementation

STARS-RC-00990/001/00
Publication No. GR-7670-1163(NP)
25 October 1990

Data Type: A005, Informal Technical Data

CONTRACT NO. F19628-88-D-0031
Delivery Order 0002

Prepared for:

Electronic Systems Division
Air Force Systems Command, USAF
Hanscom AFB, MA 01731-5000

Prepared by:

Unisys Defense Systems
Tactical Systems Division
12010 Sunrise Valley Drive
Reston, VA 22091

PREFACE

This document was prepared by Unisys Corporation, Valley Forge Laboratories, in support of the Unisys STARS Prime contract under the Process/Environment Integration task (UR20). This CDRL, 00990, is type A005 (Informal Technical Data) and is entitled "Ada Command Environment (ACE) Version 8.0, Version Description Document".

Reviewed by: [illegible]
Teri F. Payton, System Architect

Approved by: [illegible]
Hans W. Polzer, Program Manager

Contents

1 SCOPE

1.1 Identification

Version Description Document,
Ada Command Environment (ACE),
Version 8.0,
SunOS Implementation

1.2 System Overview

A set of Ada abstract data types (ADTs) is the underlying substrate that defines a common, Ada-oriented interface to diverse host environments. The Ada ADTs, in conjunction with the use of Ada as a command language, serve as a unifying concept in the description of a portable Ada command environment. The benefits provided by ADTs and Ada in a software engineering environment are extended into the command language arena. The Ada Command Environment combines the power of Ada as a command language with the description of the host environment through ADTs. ACE presents to the user a consistent Ada-oriented, development environment that supports a uniform interface across a heterogeneous set of development architectures.

This document provides an overview for version 8.0 of the Ada Command Environment (ACE) software. This software was developed in its original form under the STARS Foundations program, administered by the Office of Naval Research. This distribution includes modifications to the ACE software in several key areas, which are briefly described below.

2 RELATED SOFTWARE

The X Window System, Version 11 Release 3 (X11R3), available through the MIT/X Consortium.

The Unisys STARS Ada/Xlib bindings (from Version 2 of the Unisys Ada/Xt Toolkit), available from Unisys STARSCenter.

Sun-3/Unix CAIS-A Implementation, Version 4.5.3, available from Unisys STARSCenter.

3 VERSION DESCRIPTION

3.1 Inventory of Contents

The ACE distribution is structured as shown below. The top-level directory **ace** includes PostScript (**VDDace.ps**) and clear ASCII text (**VDDace.tty**) versions of this document, along with a complete directory listing of the ACE distribution (**Contents.tty**, reproduced herein as **Appendix A**).

```
ace
ace/code
ace/code/common
ace/code/common/vads
ace/code/common/telesoft
ace/code/design
ace/code/design/vads
ace/code/design/telesoft
ace/code/src
ace/code/src/vads
ace/code/src/C
ace/code/src/Parser_specs
ace/code/src/Scanner_specs
ace/code/src/telesoft
ace/bin
ace/docs
ace/startups
```

3.1.1 Subdirectory: ace/code

The **code** subdirectory contains C shell scripts and associated support files that are needed to build ACE from this distribution. The Ada source code for ACE is divided into the following **code** subdirectories:

3.1.1.1 Subdirectory: ace/code/common. The **code/common** subdirectory contains reusable Ada code (specifications and implementations) which can be used independently of ACE. Included in this directory, in addition to several commonly available packages, is a key binding facility that can be used to provide any Ada application with command line editing and history functions.

3.1.1.2 Subdirectory: ace/code/design. The **code/design** subdirectory contains ACE-specific Ada specifications.

3.1.1.2.1 Subdirectory: ace/code/design/telesoft. This subdirectory contains TeleSoft-specific versions of certain Ada specifications.

3.1.1.2.2 Subdirectory: ace/code/design/vads.

This subdirectory contains VADS-specific versions of certain Ada specifications.

3.1.1.3 Subdirectory: ace/code/src.

The **code/src** subdirectory contains ACE-specific Ada implementations. In addition, three subdirectories contain non-Ada source code and grammar definitions for ACE's lexical scanner and parsers.

3.1.1.3.1 Subdirectory: ace/code/src/telesoft.

This subdirectory contains TeleSoft-specific versions of certain Ada implementations.

3.1.1.3.2 Subdirectory: ace/code/src/vads.

This subdirectory contains VADS-specific versions of certain Ada implementations.

3.1.1.3.3 Subdirectory: ace/code/src/C.

This subdirectory contains "C" support files.

3.1.1.3.4 Subdirectory: ace/code/src/Scanner_specs.

This subdirectory contains the specifications processed by the **adalex** lexical scanner generator to produce the ACE lexical scanner implementation in the source files listed below. This information is provided for reference and is not needed for installation or operation of the ACE distribution.

lex.a

3.1.1.3.5 Subdirectory: ace/code/src/Parser_specs.

This subdirectory contains the specifications processed by the **ayacc** parser generator to produce the ACE parser implementation in the source code files listed below. This information is provided for reference and is not needed for installation or operation of this ACE distribution.

compilation_unit.a

decl_part.a

seq_of_stmts.a

3.1.2 Subdirectory: /bf ace/bin

The **bin** subdirectory contains Unix C shell scripts that are used by ACE to execute operating system functions and aid in window placement. The user must modify these scripts to reflect the actual operating environment.

3.1.3 Subdirectory: ace/startups

This subdirectory contains the interpreted Ada code necessary to define the initial Ada environment provided by ACE.

startup.ace - This file should be copied to the user's home directory. It is the file that is read when ACE is invoked, and it specifies what startup environment that user will have.

acvc.ace - Contains routines necessary to execute the ACVC 1.10 test suite.

commands.ace - Contains routines that are analogous to commands normally found in a command environment (e.g., directory list, edit, compile).

cpu_time.ace - Defines routines to record and report CPU time

developer.ace - Defines routines hat ACE developers use (this is not a static set of routines).

startup40.ace - Release 4.0's startup.ace file.

standard.ace - Defines standard Ada packages (e.g., Text_Io)

windowing.ace - Defines windowing operations.

cais.ace - Defines CAIS-A tools and operations.

bindings.ace - Defines Key mapping functions.

xt.ace - Defines the ACE interface to the X Window System.

3.1.4 Subdirectory: docs

The **docs** subdirectory contains PostScript (**.ps**) and clear ASCII text (**.tty**) versions of the ACE user's manual.

A complete list of files is contained in Appendix A.

3.2 Changes Installed

3.2.1 Xt Toolkit Support

The major enhancement in Version 8.0 over Version 6.0 (Version 7.0 was an internal release) is its extensive support for the X Window System's Xt toolkit. Ada interfaces are provided to the complete Hewlett-Packard widget set, as well as many of the Xt intrinsic functions. This makes it possible to prototype Xt widget applications with ACE.

Files that were added to support Xt prototyping are:

```
src/ace_hp_widgets.a
src/ace_widget.a
src/ace_x_windows.a
design/ace_hp_widgets.a
design/ace_widget.a
design/ace_x_windows.a
common/hp_widgets.a
common/hp_widgets_.a
common/intrinsics.a
common/renamed_xlib_types.a
common/stringdefs.a
common/widget.a
common/widget_.a
src/C/xwc.c
startups/xt.ace
```

Files that were modified to support Xt prototyping are:

```
src/ebuilt.a
src/ace_adt.a
src/system.a
src/C/cflush.c
src/C/xws.c
design/system.a
startups/commands.ace
startups/standard.ace
```

3.2.2 Command Histories

Several enhancements were made to the command history code, including the ability to call the ACE interpreter from within a key binding. This required modification of:

```
common/sinput.a
common/sinput_.a
common/tty_ops.a
common/tty_ops_.a
common/keymap.a
common/keymap_.a
src/ebuilt.a
startups/bindings.ace
```

Changing the package **Sanctified_Input** into a generic caused the addition of:

```
design/ace_input.a
src/ace_input.a
```

3.2.3 Static Semantics Support

Support for processing the static semantics of **Array** declarations and references required the modification of:

```
src/cexpr.a
src/semant.a
src/smgmt.a
src/compilation_unit.a
src/decl_part.a
src/seq_of_stmts.a
src/Parser_specs/compilation_unit.y
src/Parser_specs/decl_part.y
src/Parser_specs/seq_of_stmts.y
design/semant.a
```

3.2.4 Expanded String Arguments

The following files were created or modified in order to support the use of expanded names in string arguments:

```
design/expand.a
src/expand.a
```

3.2.5 Miscellaneous

Many small problems were fixed. These included determining the correct size of an **out** string parameter to a built-in routine, improper exception handling during subprogram parameter elaboration, correctly identifying an expanded name, comment changes, etc. Files that were modified are:

```
src/ace_adt.a
src/euser.a
src/user.a
src/ebuilt.a
src/debug.a
src/atextio.a
src/context.a
```

```
src/drtns.a
src/dump.a
src/eprag.a
src/error.a
src/literl.a
src/misc.a
design/debug.a
design/misc.a
```

File **src/rwace.a** was moved to **design/rwace.a**.

File **src/observe_window.icn** was moved to **src/C/observe_window.icn**.

The following files were removed from the distribution, since they were not directly related to the execution of ACE:

```
bin/move-if-change
misc/sysenv.a
src/cexpand.a
src/Make.script_unix
src/Makefile.unix_example
src/C/sunws.c
```

3.3 Adaptation Data

3.3.1 Operating Environment

The ACE prototype operates on a Sun-3 workstation. ACE provides an interface to the X Window System, but may be executed independently, without a supporting windowing system. Window manipulation operations are provided with the X Window System Ada/Xlib bindings from the Unisys Ada/Xt Toolkit.

To execute ACE, the suggested configuration is a Sun-3 workstation running:

SunOS, Version 4.0.3

MIT X Window System, Version 11, Release 3 (if running ACE with X)

3.3.2 Development Environment

To create an ACE executable image, the suggested configuration is a Sun-3 workstation running:

SunOS, Version 4.0.3

MIT X Window System, Version 11, Release 3 (if running ACE with X)

Verdix Ada Development System, Version 5.5t

Ada/Xlib binding to the X Window System, from Version 2 of the Unisys Ada/Xt Toolkit.

C compiler provided with SunOS 4.0.3

Further information on the system dependent parts of ACE and hov. to port these to other environments is given in section 4.0.3

3.3.3 Configuration-Unique Data

ACE was designed with portability as one of its primary goals. Operating and Ada compilation system dependent routines have been isolated to the **System_Dependent_Routines** package, with the minor exceptions noted below. Of the many features designed into ACE, the least portable is the interface to the host window system. Operating system dependencies, and subseq'ent symbolic debugger dependencies on the host OS, comprise the remainder of configuration-dependent features.

3.3.3.1 System_Dependent_Routines Package. The **System_Dependent_Routines** package contains the procedures which implement file system manipulations and interrupt handling (see files **system.a** and **system.cais.a**).

ACE's hierarchical file system abstract data type (ADT) and command language facilities dictate that it provide access to the host file system utilities. Porting considerations between different file systems include minor differences in naming conventions (name length and lexical character limitations) and more complex differences between how Unix and other operating systems treat directory structures. In these situations, it is best to use the host OS-supplied routines or the interfaces directly provided by the host Ada compilation system.

Interrupt handling is very operating system (and system architecture) dependent. Interrupt handling is primarily used by the ACE symbolic debugger ADT, but similar mechanisms could be used in the future to implement exception handling or Ada tasking. Depending on the host OS interface provided by the Ada compilation system, somewhat complex rework might be necessary. For example, the Ada compilation systems on the Unisys PC/IT provide an Ada interface to handle interrupts.

3.3.3.2 Ace_Universal_Types Package. Data sizes for types such as **Integer** vary significantly from one Ada compilation system to another. In order to isolate this, ACE defines its own integer type (**Ace_Integer**) with an explicit range. This forces the compilation system to choose an appropriate machine representation for the **Ace_Integer** type or inform the user of its inability to do so.

3.3.3.3 TTY_Operations Package.

The **TTY_Operations** package provides screen manipulation routines, which are necessarily configuration dependent. The current implementation of **TTY_Operations** uses **termcap**, a terminal capability database, to handle screen manipulation. For most systems there is a public domain implementation of **termcap**. If a version of **termcap** is not available for a system that is to run ACE, then the body of **TTY_Operations** would need to be rewritten to remove the references to **termcap**.

3.3.3.4 Command Language Scripts.

Other current host OS dependencies include the use of accessory native command language scripts to provide needed, high-level host dependent functionality. This is provided so users can interchange compatible host applications without recompiling ACE itself. The **ace_edit** and **ace_ada** files are two such scripts. However, the need for such scripts is defined by the environment implementor, since all the functionality provided by the scripts could also be provided in the system environment implementation (i.e., the interpreted Ada ADT bodies).

3.4 Interface Compatibility

Changes to ACE appearing in this version will not affect other components of the user's system. This is true even though this version introduces a new interface to the X Window System library.

3.5 Installation Instructions

This section contains the instructions necessary to construct ACE from the source files contained in the version 8.0 distribution. ACE executables have been successfully compiled using the Verdix and TeleSoft compilation systems. The Verdix compilation is the recommended ruggedized version. The following configurations are supported:

Verdix, version 5.5t, on SunOS 4.0.3

- with/without X11R3 support
- with/without CAIS-A support

TeleSoft, version 1.4, on SunOS 4.0.3

- with/without X11R3 support
- without CAIS-A support

Currently, only the Verdix version of ACE supports both CAIS-A interaction and stand-alone operation. All other versions of ACE are stand-alone. Only the source code for ACE is included in this delivery. Contact Unisys STARSCenter for the Ada/X Window System Xlib bindings and/or CAIS-A distributions.

3.5.1 VADS Build Procedure

1. Edit the environment variables in file **Build_ACE.var** to reflect the actual operating environment. The following environment variables must be modified:

   VADS_BASE (Optional) - identifies the full pathname of the VADS compilation system (e.g., /mybase/compilers/vads_5.5). Needed only if building with the VADS compilation system.

   TELEGEN2 (Optional) - identifies the full pathname of the TeleSoft compilation system (e.g., /mybase/compilers/telegen_1.4). Needed only if building with the TeleSoft compilation system.

   ADA_XLIB (Optional) - identifies the full pathname of the Ada libraries for the Ada/Xt Toolkit Xlib bindings (e.g., /mybase/adaxt/code/Xlib). Needed only if building for operation with an X Window System interface.

   CAIS_LIB (Optional) - identifies the full pathname of the nonshared portion of the CAIS-A installation (e.g., /mybase/cais-a/src/nonshared). Needed only if you are compiling for operation under the CAIS-A object management system.

   LIB_X_SUPPORT (Optional) - identifies the full pathname of the standard C utility library from the X Window System distribution or from the Ada/Xt Toolkit (e.g., /mybase/adaxt/code/C/lib.a). Needed only if building for operation with an X Window System interface.

   ACE - identifies the full pathname of the top-level ACE directory (e.g., /mybase/ace).

   If the $ACE directory is structured as described in this document, no further modifications are necessary. If not, the following additional variables in **Build_ACE.var** will have to be modified to indicate which host directory contains each of the major code components of this release:

   ACE_CODE

   ACE_COMMON

   ACE_DESIGN

   ACE_SRC

   ACE_CLIB

   ACE_BIN

   LOG

Other environment variables may need to change, depending on the installer's system configuration. A complete listing of the **Build_ACE.var** file is included in Appendix B.

2. Execute **Build_ACE.VADS**, or **Build_ACE.TeleSoft**, as appropriate, providing configuration information when prompted by the script.

3.5.2 Startup Files

There are several startup files provided with this distribution (see the **startups** directory for their source). Users may (and should) configure their individual **startup.ace** file to suit their needs. The **startup.ace** file should reside in a user's home directory. The following code is an example **startup.ace**:

```
--pragma echo(on);

-- get routines for measuring CPU.
Interpret_File ("/ace/startups/cpu_time.ace");

-- Variables for clocking our startup speed.

Start : Time;
Stop  : Time;

 -- Start ticking
 Start := Clock;

 Interpret_File ("/ace/startups/standard.ace");
 Interpret_File ("/ace/startups/commands.ace");
 Interpret_File ("/ace/startups/windowing.ace");
 Interpret_File ("/ace/startups/bindings.ace");

 -- Stop Ticking
 Stop := Clock;

 -- How much time?

 Put ("Startup CPU seconds: ");
 Put_Time(Difference(Stop, Start));

 -- ASCII Terminal clear to EOL.
 Put_Line(Ascii.Esc & "[K");
```

3.5.3 Execution of ACE

Type **ACE** at the shell prompt to invoke ACE stand-alone, where **ACE** is the name of the executable made from the compiling and linking steps described above. ACE is invoked automatically from CAIS-A as the user's login shell. Upon startup, the stand-alone ACE reads the startup file from the user's home directory, while the CAIS-A version will follow the rules specified in the CAIS-A script for starting ACE. To install the new ACE executable and **startup.ace** into the CAIS-A baseline SEE, replace the existing ACE executable (in **Baseline_SEE/cais-a/bin/ACE.exec**) and the existing ACE startup file (in **Baseline_SEE/cais-a/lib/startup.ace**). See the Unisys STARS Baseline SEE, Virtual Interface Implementation 3, Informal Report, 26 September 1989 for more information.

3.6 Potential Problems

Several problems are known to exist in the ACE 8.0 distribution:

1. Non-terminating recursive functions may cause ACE to lose track of declared objects. Currently, there is no known workaround (other than not writing non-terminating functions).
2. Calling the **Ace_Adt.Interpret** procedure recursively is illegal and can cause ACE to terminate unexpectedly; sometimes this can occur without a direct call by the user, i.e., calling **Edit_And_Interpret** during the interruption of an X callback.
3. Several enumeration values of **Key_Bindings.Commands** will have no effect when used in the **Make_Bindings** command. They are:

   ```
   Get_Current_Line
   Get_Current_Character
   Get_Current_Column
   ```

   They will be removed from the enumeration type in the next release. They cannot be removed in the current version, since there exists an underlying dependency in the **Evaluate_Built_In_Subprogram** subprogram.
4. A function cannot be used as one of the bounds of a **String** object declaration; the workaround is to assign the function value to an object and use the object in the **String** object declaration.

3.7 Enhancements

Ultimately, ACE will be an interpreter for the entire Ada language. This will provide the foundation for a portable interactive environment for Ada coding. By extending the interpreter with complete support for the X Window System, an environment similar to InterLisp and Smalltalk could be created for Ada, which would remove many of the barriers that currently exist to a rapid prototyping Ada system.

4 NOTES

A Appendix: Inventory of Contents

```
NOTE:  "*" identifies executables; "/" identifies directories

ace:
Contents.tty
VDDace.ps
VDDace.tty
bin/
code/
docs/
startups/

ace/bin:
ace_ada*
ace_edit*
place_observe_window.csh*

ace/code:
Build_ACE.TeleSoft*
Build_ACE.VADS*
Build_ACE.var
common/
design/
src/

ace/code/common:
btrees.a
btrees_.a
hash_table.a
hash_table_.a
hp_widgets.a
intrinsics.a
keymap.a
keymap_.a
lists.a
lists_.a
renamed_xlib_types.a
sinput.a
sinput_.a
stringdefs.a
telesoft/
tty_ops.a
unix_types.a
vads/
```

```
widget.a

ace/code/common/telesoft:
common.alb
common.x.alb
hp_widgets_.a
tty_ops_.a
unix_types_.a
widget_.a

ace/code/common/vads:
hp_widgets_.a
tty_ops_.a
unix_types_.a
widget_.a

ace/code/design:
ace_adt.a
ace_hp_widgets.a
ace_input.a
ace_widget.a
ace_x_windows.a
amain.a
aprgut.a
astd.a
atextio.a
auntyp.a
calendar.a
common_parser.a
compilation_unit.a
context.a
cpu.a
create.a
ctree.a
debug.a
decl_part.a
dir.a
drsup.a
drtns.a
dspprt.a
dsubsup.a
dump.a
error.a
expand.a
files.a
```

```
get.a
help.a
lex.a
lexact.a
literl.a
misc.a
miscs.a
obj.a
ooe.a
preprs.a
prsdef.a
rwace.a
sdb.a
semant.a
seq_of_stmts.a
set.a
smgms.a
smgmt.a
stget.a
stmtev.a
string.a
strt.a
sts.a
stset.a
system_environment_.a
telesoft/
tokens_definition.a
treeb.a
user.a
vads/
wndobj.a
ws.a
xadt.a
yyerr.a

ace/code/design/telesoft:
design.alb
design.x.alb
sunview.a
system.a
x.a

ace/code/design/vads:
sunview.a
system.a
```

```
x.a

ace/code/src:
C/
Parser_specs/
Scanner_specs/
ace_adt.a
ace_hp_widgets.a
ace_input.a
ace_widget.a
ace_x_windows.a
alloc.a
amain.a
aprgut.a
astd.a
atextio.a
calendar.a
carray.a
cattr.a
cdot.a
cexpr.a
cinfix.a
cmsppt.a
common_parser.a
compilation_unit.a
compilation_unit_goto.a
compilation_unit_shift_reduce.a
context.a
create.a
csmisc.a
cstmt.a
csubpgm.a
ctree.a
debug.a
decl_part.a
decl_part_goto.a
decl_part_shift_reduce.a
denum.a
dir.a
dpkg.a
drecs.a
drsup.a
drtns.a
dspprt.a
dsubprg.a
```

```
dsubsup.a
dtyped.a
dump.a
eattr.a
eblock.a
ebuilt.a
ebuilt.cais.a
ebuilt.x.a
ebuilt.xcais.a
eexpr.a
einfix.a
eistmt.a
eobject.a
epkgb.a
eprag.a
error.a
estmt.a
esubprm.a
euser.a
expand.a
files.a
get.a
help.a
initsym.a
lex.a
lexact.a
literl.a
main.a
misc.a
miscs.a
nowin.a
ooe.a
pre_parser.a
rwace.a
semant.a
seq_of_stmts.a
seq_of_stmts_goto.a
seq_of_stmts_shift_reduce.a
set.a
smgms.a
smgmt.a
staddpk.a
stget.a
stmtecs.a
stmtev.a
```

```
strt.a
stset.a
telesoft/
user.a
vads/
xadt.a
yyerr.a

ace/code/src/C:
cflush.c
observe_window.icn
sunws.c
sys_env.c
xwc.c
xws.c

ace/code/src/Parser_specs:
compilation_unit.y
decl_part.y
seq_of_stmts.y

ace/code/src/Scanner_specs:
scan_adalex.l

ace/code/src/telesoft:
onlyx.a
src.alb
src.cais.alb
src.x.alb
src.xcais.alb
sunview.a
system.a
system.cais.a
system_environment.a
ws.a
x.a

ace/code/src/vads:
onlyx.a
sunview.a
system.a
system.cais.a
system_environment.a
ws.a
x.a
```

```
ace/docs:
UserManual.ps
UserManual.tty

ace/startups:
acvc.ace
bindings.ace
cais.ace
commands.ace
cpu_time.ace
developer.ace
standard.ace
startup.ace
startup40.ace
windowing.ace
xt.ace
```

B Appendix: Build Scripts

B.1 File: Build_ACE.var

```
#
# Establish a path to the VADS compilation system.
#
setenv VADS_BASE     <path to the VADS compilation system
                      (e.g. /mybase/compilers/vads5.5)>
setenv VADS_BIN      $VADS_BASE/bin
set    path       =  ( $VADS_BIN $path )
setenv ADA           " ada -w -O0 "

 #
 # Establish a path to the TeleSoft compilation system.
 #
 setenv TELEGEN2      <path to the TeleSoft compilation system
                       (e.g., /mybase/compilers/telegen_1.4)>
 setenv TADA          " $TELEGEN2/bin/ada -v "
 setenv TALD          " $TELEGEN2/bin/ald -v -V 2000 "
 setenv TACR          " $TELEGEN2/bin/acr -f -m 32000 "

 #
 # Define C Languuage compilation variable
 #
 setenv CC            " cc -g -c "

 #
 # Define the location of the required source code directories
 # and X11R3 C archive
 #
 setenv ADA_XLIB      <path to the Ada libraries for the Ada/Xt Toolkit
                       Xlib bindings (e.g. /mybase/adaxt/code/Xlib)>

 setenv CAIS_LIB      <path to the Ada libraries for the nonshared portion of
                       CAIS-A (e.g. /mybase/cais-a/src/nonshared)>

 setenv LIB_X_SUPPORT <path to the standard C utility library from the X Window
                       System distribution or from the Ada/Xt Toolkit
                       (e.g. /mybase/adaxt/code/C/lib.a)>

 setenv ACE           <path to the top-level ACE directory (e.g. /mybase/ace)>
 setenv ACE_CODE      $ACE/code
 setenv ACE_COMMON    $ACE_CODE/common
 setenv ACE_DESIGN    $ACE_CODE/design
setenv ACE_SRC       $ACE_CODE/src
setenv ACE_CLIB      $ACE_CODE/src/C

#
# Define the location of C support utilities for the X Window interface
#
setenv LIB_MISC           $ACE_CLIB/cflush.o
setenv LIB_ARG            $ACE_CLIB/sys_env.o
setenv LIB_WIDGETCLASSES  $ACE_CLIB/xwc.o
setenv LIB_WINDOW_SUPPORT $ACE_CLIB/xws.o

#
# The following variables locate various components of the X11R3 distribution.
# The pathnames shown are typical, but can vary from one installation to
# another.  Please consult your system administrator.
#
setenv LIB_HP_WIDGETS     /usr/lib/X11R3/libXw.a
setenv LIB_ATHENA_WIDGETS /usr/lib/X11R3/libXaw.a
setenv LIB_XT             /usr/lib/X11R3/libXt.a
setenv LIB_MU             /usr/lib/X11R3/libXmu.a
setenv LIB_X11            /usr/lib/X11R3/libX11.a
setenv LIB_RESOLV         /usr/lib/libresolv.a
setenv LIB_TERMCAP        /usr/lib/libtermcap.a

setenv LIBRARIES_WITH_X   " $LIB_MISC              \
                          $LIB_X_SUPPORT           \
                          $LIB_WIDGETCLASSES       \
                          $LIB_WINDOW_SUPPORT      \
                          $LIB_HP_WIDGETS          \
                          $LIB_ATHENA_WIDGETS      \
                          $LIB_XT                  \
                          $LIB_MU                  \
                          $LIB_X11                 \
                          $LIB_RESOLV              \
                          $LIB_TERMCAP "

setenv LIBRARIES_NO_X     " $LIB_MISC $LIB_RESOLV $LIB_TERMCAP "
```

B.2 Script: Build_ACE.VADS

```
#! /bin/csh -f
echo ""
echo "Defining installation-dependent variables"
echo ""
source Build_ACE.var

setenv TARGET $ACE_CODE/Build_VADS
setenv LOG    $TARGET/Build_ACE.Log

 #
 set Caisop="n"
 set Xop="n"
 echo -n "Are you building for operation under CAIS-A? [y n](n) "
 set Caisop=$<
 if ( $Caisop == "y" || $Caisop == "Y" ) then
  set Caisop="y"
  echo "Building for operation under CAIS-A.  Thank you."
 else
  set Caisop="n"
  echo "Building for stand-alone operation.  Thank you."
 endif
 echo ""

 echo -n "Are you building for operation with an X Window System interface? [y n](n) "
 set Xop=$<
 if ( $Xop == "y" || $Xop == "Y" ) then
  set Xop="y"
  echo "Building for operation with an X Window System interface.  Thank you."
 else
  set Xop="n"
  echo "Building for operation without an X Window System interface.  Thank you."
 endif
 echo ""

 #
 if ( ! -d $TARGET ) mkdir $TARGET
 if ( ! -d $TARGET/common ) mkdir $TARGET/common
 if ( ! -d $TARGET/design ) mkdir $TARGET/design
 if ( ! -d $TARGET/src ) mkdir $TARGET/src
 if ( ! -d $TARGET/bin ) mkdir $TARGET/bin

 #
 echo "Building Ada libraries in each sub-directory"
echo ""
#
foreach dir (common design src)
  a.mklib -f $TARGET/$dir $VADS_BASE/verdixlib
end

#
echo "Establishing dependencies"
echo ""
#
cd $TARGET/common
if ( $Xop == "y" ) then
  a.path -a $ADA_XLIB
endif

cd $TARGET/design
a.path -a $TARGET/common
if ( $Xop == "y" ) then
  a.path -a $ADA_XLIB
endif
a.path -a $TARGET/src

cd $TARGET/src
a.path -a $TARGET/design
if ( $Xop == "y" ) then
  a.path -a $ADA_XLIB
endif
a.path -a $TARGET/common

if ( $Caisop == "y" ) then
 a.path -a $CAIS_LIB
endif

#
echo "Creating source code links in $TARGET"
echo ""
cd $TARGET/common
foreach file ($ACE_COMMON/*.a)
  ln -s $file ${file:t}
end
foreach file ($ACE_COMMON/vads/*.a)
  ln -s $file ${file:t}
end

cd $TARGET/design
```

```
foreach file ($ACE_DESIGN/*.a)
  ln -s $file ${file:t}
end
foreach file ($ACE_DESIGN/vads/*.a)
  ln -s $file ${file:t}
end

cd $TARGET/src
foreach file ($ACE_SRC/*.a)
  ln -s $file ${file:t}
end
foreach file ($ACE_SRC/vads/*.a)
  ln -s $file ${file:t}
end

#
echo "Initializing the build log, file $LOG"
echo ""
#
if -e $LOG rm -f $LOG

#
echo "Compiling the code"
echo ""
#
echo "Subdirectory:  common"
echo ""
#
cd $TARGET/common
    date                            >>& $LOG

    $ADA unix_types.a               >>& $LOG
    $ADA unix_types_.a              >>& $LOG
    $ADA tty_ops.a                  >>& $LOG
    $ADA tty_ops_.a                 >>& $LOG
    $ADA lists.a                    >>& $LOG
    $ADA lists_.a                   >>& $LOG
    $ADA hash_table.a               >>& $LOG
    $ADA hash_table_.a              >>& $LOG
    $ADA btrees.a                   >>& $LOG
    $ADA btrees_.a                  >>& $LOG
    $ADA keymap.a                   >>& $LOG
    $ADA keymap_.a                  >>& $LOG
    $ADA sinput.a                   >>& $LOG
    $ADA sinput_.a                  >>& $LOG
```

```

if ( $Xop == "y" ) then
    $ADA renamed_xlib_types.a     >>& $LOG
    $ADA stringdefs.a             >>& $LOG
    $ADA intrinsics.a             >>& $LOG
    $ADA widget.a                 >>& $LOG
    $ADA widget_.a                >>& $LOG
    $ADA hp_widgets.a             >>& $LOG
    $ADA hp_widgets_.a            >>& $LOG
endif

#
echo "Subdirectory:  design"
echo ""
#
cd $TARGET/design
    date                          >>& $LOG

    $ADA auntyp.a                 >>& $LOG
    $ADA misc.a                   >>& $LOG
    $ADA lex.a                    >>& $LOG
    $ADA ws.a                     >>& $LOG
    $ADA literl.a                 >>& $LOG

#
echo "filename src/literl.a"
echo ""
#
cd $TARGET/src
    $ADA literl.a                 >>& $LOG

#
echo "Continuing with subdirectory:  design"
echo ""
#
cd $TARGET/design
    $ADA string.a                 >>& $LOG
    $ADA smgms.a                  >>& $LOG
    $ADA sunview.a                >>& $LOG
    $ADA atextio.a                >>& $LOG
    $ADA ooe.a                    >>& $LOG
    $ADA files.a                  >>& $LOG
    $ADA user.a                   >>& $LOG
    $ADA dir.a                    >>& $LOG

```

```
if ( $Xop == "n" ) then
    $ADA system_environment_.a      >>& $LOG
endif

    $ADA system.a                   >>& $LOG
    $ADA help.a                     >>& $LOG
    $ADA sts.a                      >>& $LOG
    $ADA sdb.a                      >>& $LOG
    $ADA context.a                  >>& $LOG
    $ADA amain.a                    >>& $LOG
    $ADA treeb.a                    >>& $LOG
    $ADA stmtev.a                   >>& $LOG
    $ADA debug.a                    >>& $LOG
    $ADA stset.a                    >>& $LOG
    $ADA stget.a                    >>& $LOG
    $ADA preprs.a                   >>& $LOG
    $ADA ctree.a                    >>& $LOG
    $ADA dspprt.a                   >>& $LOG
    $ADA create.a                   >>& $LOG
    $ADA miscs.a                    >>& $LOG
    $ADA drtns.a                    >>& $LOG
    $ADA prsdef.a                   >>& $LOG
    $ADA yyerr.a                    >>& $LOG
    $ADA tokens_definition.a        >>& $LOG
    $ADA common_parser.a            >>& $LOG
    $ADA seq_of_stmts.a             >>& $LOG
    $ADA get.a                      >>& $LOG
    $ADA set.a                      >>& $LOG
    $ADA ace_adt.a                  >>& $LOG
    $ADA dump.a                     >>& $LOG
    $ADA dsubsup.a                  >>& $LOG
    $ADA semant.a                   >>& $LOG
    $ADA smgmt.a                    >>& $LOG
    $ADA strt.a                     >>& $LOG
    $ADA error.a                    >>& $LOG
    $ADA compilation_unit.a         >>& $LOG
    $ADA decl_part.a                >>& $LOG
    $ADA drsup.a                    >>& $LOG
    $ADA x.a                        >>& $LOG
    $ADA lexact.a                   >>& $LOG
    $ADA calendar.a                 >>& $LOG
    $ADA obj.a                      >>& $LOG
    $ADA wndobj.a                   >>& $LOG
    $ADA xadt.a                     >>& $LOG
    $ADA astd.a                     >>& $LOG
```

```
        $ADA aprgut.a                 >>& $LOG
        $ADA rwace.a                  >>& $LOG
        $ADA expand.a                 >>& $LOG

if ( $Xop == "y" ) then
        $ADA ace_x_windows.a          >>& $LOG
        $ADA ace_widget.a             >>& $LOG
        $ADA ace_hp_widgets.a         >>& $LOG
endif

        $ADA ace_input.a              >>& $LOG

#
echo "Subdirectory:  src"
echo ""
#
cd $TARGET/src
        date                          >>& $LOG

        $ADA atextio.a                >>& $LOG
        $ADA context.a                >>& $LOG
        $ADA dir.a                    >>& $LOG
        $ADA main.a                   >>& $LOG
        $ADA user.a                   >>& $LOG
        $ADA ooe.a                    >>& $LOG
        $ADA miscs.a                  >>& $LOG
        $ADA debug.a                  >>& $LOG
        $ADA dump.a                   >>& $LOG
        $ADA dspprt.a                 >>& $LOG
        $ADA smgmt.a                  >>& $LOG
        $ADA stmtev.a                 >>& $LOG
        $ADA estmt.a                  >>& $LOG
        $ADA eistmt.a                 >>& $LOG
        $ADA stmtecs.a                >>& $LOG
        $ADA esubprm.a                >>& $LOG
        $ADA eexpr.a                  >>& $LOG
        $ADA eattr.a                  >>& $LOG
        $ADA amain.a                  >>& $LOG
        $ADA set.a                    >>& $LOG
        $ADA create.a                 >>& $LOG
        $ADA get.a                    >>& $LOG
        $ADA stset.a                  >>& $LOG
        $ADA stget.a                  >>& $LOG
        $ADA help.a                   >>& $LOG
        $ADA semant.a                 >>& $LOG
```

```
    $ADA cexpr.a                  >>& $LOG
    $ADA cattr.a                  >>& $LOG
    $ADA csubpgm.a                >>& $LOG
    $ADA csmisc.a                 >>& $LOG
    $ADA cmsppt.a                 >>& $LOG

if ( $Xop == "n" ) then
    $ADA system_environment.a     >>& $LOG
endif

if ( $Caisop == "y" ) then
    $ADA system.cais.a            >>& $LOG
else
    $ADA system.a                 >>& $LOG
endif

    $ADA files.a                  >>& $LOG
    $ADA cstmt.a                  >>& $LOG
    $ADA drtns.a                  >>& $LOG
    $ADA denum.a                  >>& $LOG
    $ADA drecs.a                  >>& $LOG
    $ADA dpkg.a                   >>& $LOG
    $ADA dsubprg.a                >>& $LOG
    $ADA error.a                  >>& $LOG
    $ADA strt.a                   >>& $LOG
    $ADA staddpk.a                >>& $LOG
    $ADA initsym.a                >>& $LOG
    $ADA ace_adt.a                >>& $LOG
    $ADA dsubsup.a                >>& $LOG
    $ADA drsup.a                  >>& $LOG
    $ADA x.a                      >>& $LOG
    $ADA lexact.a                 >>& $LOG
    $ADA yyerr.a                  >>& $LOG
    $ADA lex.a                    >>& $LOG
    $ADA misc.a                   >>& $LOG
    $ADA ctree.a                  >>& $LOG
    $ADA common_parser.a          >>& $LOG
    $ADA decl_part_shift_reduce.a >>& $LOG
    $ADA decl_part_goto.a         >>& $LOG
    $ADA decl_part.a              >>& $LOG
    $ADA pre_parser.a             >>& $LOG
    $ADA compilation_unit_shift_reduce.a >>& $LOG
    $ADA compilation_unit_goto.a   >>& $LOG
    $ADA compilation_unit.a         >>& $LOG
    $ADA seq_of_stmts_shift_reduce.a >>& $LOG
    $ADA seq_of_stmts_goto.a      >>& $LOG
    $ADA seq_of_stmts.a           >>& $LOG
    $ADA calendar.a               >>& $LOG
    $ADA main.a                   >>& $LOG
    $ADA rwace.a                  >>& $LOG
    $ADA astd.a                   >>& $LOG
    $ADA expand.a                 >>& $LOG
    $ADA aprgut.a                 >>& $LOG
    $ADA alloc.a                  >>& $LOG
    $ADA cinfix.a                 >>& $LOG
    $ADA cdot.a                   >>& $LOG
    $ADA carray.a                 >>& $LOG

if (( $Caisop == "y" ) && ( $Xop == "y" )) then
    $ADA ebuilt.xcais.a           >>& $LOG  # ACE under CAIS-A with X
else if ( $Xop == "y" ) then
    $ADA ebuilt.x.a               >>& $LOG  # ACE under Unix with X
else if ( $Caisop == "y" ) then
    $ADA ebuilt.cais.a            >>& $LOG  # ACE under CAIS-A without X
else
    $ADA ebuilt.a                 >>& $LOG  # ACE under Unix without X
endif

    $ADA einfix.a                 >>& $LOG
    $ADA euser.a                  >>& $LOG
    $ADA eprag.a                  >>& $LOG
    $ADA eblock.a                 >>& $LOG
    $ADA epkgb.a                  >>& $LOG
    $ADA dtyped.a                 >>& $LOG

if ( $Xop == "y" ) then
    $ADA onlyx.a                  >>& $LOG
else
    $ADA nowin.a                  >>& $LOG
endif

    $ADA eobject.a                >>& $LOG

if ( $Xop == "y" ) then
    $ADA xadt.a                   >>& $LOG
endif

    $ADA smgms.a                  >>& $LOG
    $ADA ace_input.a              >>& $LOG

```

```
if ( $Xop == "y" ) then
    $ADA ace_x_windows.a            >>& $LOG
    $ADA ace_widget.a               >>& $LOG
    $ADA ace_hp_widgets.a           >>& $LOG
endif

#
echo "Finally, the C subroutines"
echo ""
#
cd $ACE_CLIB
    date                            >>& $LOG

    $CC cflush.c                    >>& $LOG

    $CC sys_env.c                   >>& $LOG

if ( $Xop == "y" ) then
    $CC xwc.c                       >>& $LOG
    $CC xws.c                       >>& $LOG
endif

#
echo "Linking the objects"
echo ""
#
cd $TARGET/src
    date                            >>& $LOG

if ( $Xop == "y" ) then
    a.ld main -o ACE $LIBRARIES_WITH_X   >>& $LOG
else
    a.ld main -o ACE $LIBRARIES_NO_X     >>& $LOG
endif

    date                            >>& $LOG

#
echo "Placing executable in $TARGET/bin"
echo ""
#
    mv -f ACE $TARGET/bin           >>& $LOG

#
echo "Build complete"
echo ""
```

B.3 Script: Build_ACE.TeleSoft

```
#! /bin/csh -f
echo ""
echo "Defining installation-dependent variables"
echo ""
source Build_ACE.var

setenv TARGET $ACE_CODE/Build_TeleSoft
setenv LOG    $TARGET/Build_ACE.Log

 #
 set Caisop="n"
 set Xop="n"
 echo -n "Are you building for operation under CAIS-A? [y n](n) "
 set Caisop=$<
 if ( $Caisop == "y" || $Caisop == "Y" ) then
  set Caisop="y"
  echo "Building for operation under CAIS-A.  Thank you."
 else
  set Caisop="n"
  echo "Building for stand-alone operation.  Thank you."
 endif
 echo ""

 echo -n "Are you building for operation with an X Window System interface? [y n](n) "
 set Xop=$<
 if ( $Xop == "y" || $Xop == "Y" ) then
  set Xop="y"
  echo "Building for operation with an X Window System interface.  Thank you."
 else
  set Xop="n"
  echo "Building for operation without an X Window System interface.  Thank you."
 endif
 echo ""

 #
 if ( ! -d $TARGET ) mkdir $TARGET
 if ( ! -d $TARGET/common ) mkdir $TARGET/common
 if ( ! -d $TARGET/design ) mkdir $TARGET/design
 if ( ! -d $TARGET/src ) mkdir $TARGET/src
 if ( ! -d $TARGET/bin ) mkdir $TARGET/bin

 #
 echo "Building Ada libraries in each sub-directory"
echo ""
#
foreach dir (common design src)
  cd $TARGET/$dir
  $TACR $dir
end

#
echo "Establishing dependencies"
echo ""
#
cd $TARGET/common
if ( $Xop == "y" ) then
  ln -s $ACE_COMMON/telesoft/common.x.alb liblst.alb
else
  ln -s $ACE_COMMON/telesoft/common.alb liblst.alb
endif

cd $TARGET/design
if ( $Xop == "y" ) then
  ln -s $ACE_DESIGN/telesoft/design.x.alb liblst.alb
else
  ln -s $ACE_DESIGN/telesoft/design.alb liblst.alb
endif

cd $TARGET/src
if ( $Xop == "y" ) then
  if ($Caisop == "y" ) then
    ln -s $ACE_SRC/telesoft/src.xcais.alb liblst.alb
  else
    ln -s $ACE_SRC/telesoft/src.x.alb liblst.alb
  endif
else if ($Caisop == "y" ) then
    ln -s $ACE_SRC/telesoft/src.cais.alb liblst.alb
  else
    ln -s $ACE_SRC/telesoft/src.alb liblst.alb
  endif
endif

#
echo "Creating source code links in $TARGET"
echo ""
cd $TARGET/common
foreach file ($ACE_COMMON/*.a)
  ln -s $file ${file:t}
```

```
end
foreach file ($ACE_COMMON/telesoft/*.a)
  ln -s $file ${file:t}
end

cd $TARGET/design
foreach file ($ACE_DESIGN/*.a)
  ln -s $file ${file:t}
end
foreach file ($ACE_DESIGN/telesoft/*.a)
  ln -s $file ${file:t}
end

cd $TARGET/src
foreach file ($ACE_SRC/*.a)
  ln -s $file ${file:t}
end
foreach file ($ACE_SRC/telesoft/*.a)
  ln -s $file ${file:t}
end

#
echo "Initializing the build log, file $LOG"
echo ""
#
if -e $LOG rm -f $LOG

#
echo "Compiling the code"
echo ""
#
echo "Subdirectory:  common"
echo ""
#
cd $TARGET/common
    date                          >>& $LOG

    $TADA unix_types.a             >>& $LOG
    $TADA unix_types_.a            >>& $LOG
    $TADA tty_ops.a                >>& $LOG
    $TADA tty_ops_.a               >>& $LOG
    $TADA lists.a                  >>& $LOG
    $TADA lists_.a                 >>& $LOG
    $TADA hash_table.a             >>& $LOG
    $TADA hash_table_.a            >>& $LOG
    $TADA btrees.a                   >>& $LOG
    $TADA btrees_.a                  >>& $LOG
    $TADA keymap.a                   >>& $LOG
    $TADA keymap_.a                  >>& $LOG
    $TADA sinput.a                   >>& $LOG
    $TADA sinput_.a                  >>& $LOG

if ( $Xop == "y" ) then
    $TADA renamed_xlib_types.a       >>& $LOG
    $TADA stringdefs.a               >>& $LOG
    $TADA intrinsics.a               >>& $LOG
    $TADA widget.a                   >>& $LOG
    $TADA widget_.a                  >>& $LOG
    $TADA hp_widgets.a               >>& $LOG
    $TADA hp_widgets_.a              >>& $LOG
endif

#
echo "Subdirectory:  design"
echo ""
#
cd $TARGET/design
    date                            >>& $LOG

    $TADA auntyp.a                   >>& $LOG
    $TADA misc.a                     >>& $LOG
    $TADA lex.a                      >>& $LOG
    $TADA ws.a                       >>& $LOG
    $TADA literl.a                   >>& $LOG

#
echo "filename src/literl.a"
echo ""
#
cd $TARGET/src
    $TADA literl.a                   >>& $LOG

#
echo "Continuing with subdirectory:  design"
echo ""
#
cd $TARGET/design
    $TADA string.a                   >>& $LOG
    $TADA smgms.a                    >>& $LOG
    $TADA sunview.a                  >>& $LOG
    $TADA atextio.a                  >>& $LOG
    $TADA ooe.a                      >>& $LOG
    $TADA files.a                    >>& $LOG
    $TADA user.a                     >>& $LOG
    $TADA dir.a                      >>& $LOG

if ( $Xop == "n" ) then
    $TADA system_environment_.a      >>& $LOG
endif

    $TADA system.a                   >>& $LOG
    $TADA help.a                     >>& $LOG
    $TADA sts.a                      >>& $LOG
    $TADA sdb.a                      >>& $LOG
    $TADA context.a                  >>& $LOG
    $TADA amain.a                    >>& $LOG
    $TADA treeb.a                    >>& $LOG
    $TADA stmtev.a                   >>& $LOG
    $TADA debug.a                    >>& $LOG
    $TADA stset.a                    >>& $LOG
    $TADA stget.a                    >>& $LOG
    $TADA preprs.a                   >>& $LOG
    $TADA ctree.a                    >>& $LOG
    $TADA dspprt.a                   >>& $LOG
    $TADA create.a                   >>& $LOG
    $TADA miscs.a                    >>& $LOG
    $TADA drtns.a                    >>& $LOG
    $TADA prsdef.a                   >>& $LOG
    $TADA yyerr.a                    >>& $LOG
    $TADA tokens_definition.a        >>& $LOG
    $TADA common_parser.a            >>& $LOG
    $TADA seq_of_stmts.a             >>& $LOG
    $TADA get.a                      >>& $LOG
    $TADA set.a                      >>& $LOG
    $TADA ace_adt.a                  >>& $LOG
    $TADA dump.a                     >>& $LOG
    $TADA dsubsup.a                  >>& $LOG
    $TADA semant.a                   >>& $LOG
    $TADA smgmt.a                    >>& $LOG
    $TADA strt.a                     >>& $LOG
    $TADA error.a                    >>& $LOG
    $TADA compilation_unit.a         >>& $LOG
    $TADA decl_part.a                >>& $LOG
    $TADA drsup.a                    >>& $LOG
    $TADA x.a                        >>& $LOG
    $TADA lexact.a                  >>& $LOG
    $TADA calendar.a                >>& $LOG
    $TADA obj.a                     >>& $LOG
    $TADA wndobj.a                  >>& $LOG
    $TADA xadt.a                    >>& $LOG
    $TADA astd.a                    >>& $LOG
    $TADA aprgut.a                  >>& $LOG
    $TADA rwace.a                   >>& $LOG
    $TADA expand.a                  >>& $LOG

if ( $Xop == "y" ) then
    $TADA ace_x_windows.a           >>& $LOG
    $TADA ace_widget.a              >>& $LOG
    $TADA ace_hp_widgets.a          >>& $LOG
endif

    $TADA ace_input.a               >>& $LOG

#
echo "Subdirectory:  src"
echo ""
#
cd $TARGET/src
    date                           >>& $LOG

    $TADA atextio.a                 >>& $LOG
    $TADA context.a                 >>& $LOG
    $TADA dir.a                     >>& $LOG
    $TADA main.a                    >>& $LOG
    $TADA user.a                    >>& $LOG
    $TADA ooe.a                     >>& $LOG
    $TADA miscs.a                   >>& $LOG
    $TADA debug.a                   >>& $LOG
    $TADA dump.a                    >>& $LOG
    $TADA dspprt.a                  >>& $LOG
    $TADA smgmt.a                   >>& $LOG
    $TADA stmtev.a                  >>& $LOG
    $TADA estmt.a                   >>& $LOG
    $TADA eistmt.a                  >>& $LOG
    $TADA stmtecs.a                 >>& $LOG
    $TADA esubprm.a                 >>& $LOG
    $TADA eexpr.a                   >>& $LOG
    $TADA eattr.a                   >>& $LOG
    $TADA amain.a                   >>& $LOG
    $TADA set.a                     >>& $LOG
    $TADA create.a               >>& $LOG
    $TADA get.a                  >>& $LOG
    $TADA stset.a                >>& $LOG
    $TADA stget.a                >>& $LOG
    $TADA help.a                 >>& $LOG
    $TADA semant.a               >>& $LOG
    $TADA cexpr.a                >>& $LOG
    $TADA cattr.a                >>& $LOG
    $TADA csubpgm.a              >>& $LOG
    $TADA csmisc.a               >>& $LOG
    $TADA cmsppt.a               >>& $LOG

if ( $Xop == "n" ) then
    $TADA system_environment.a   >>& $LOG
endif

if ( $Caisop == "y" ) then
    $TADA system.cais.a          >>& $LOG
else
    $TADA system.a               >>& $LOG
endif

    $TADA files.a                >>& $LOG
    $TADA cstmt.a                >>& $LOG
    $TADA drtns.a                >>& $LOG
    $TADA denum.a                >>& $LOG
    $TADA drecs.a                >>& $LOG
    $TADA dpkg.a                 >>& $LOG
    $TADA dsubprg.a              >>& $LOG
    $TADA error.a                >>& $LOG
    $TADA strt.a                 >>& $LOG
    $TADA staddpk.a              >>& $LOG
    $TADA initsym.a              >>& $LOG
    $TADA ace_adt.a              >>& $LOG
    $TADA dsubsup.a              >>& $LOG
    $TADA drsup.a                >>& $LOG
    $TADA x.a                    >>& $LOG
    $TADA lexact.a               >>& $LOG
    $TADA yyerr.a                >>& $LOG
    $TADA lex.a                  >>& $LOG
    $TADA misc.a                 >>& $LOG
    $TADA ctree.a                >>& $LOG
    $TADA common_parser.a        >>& $LOG
    $TADA decl_part_shift_reduce.a >>& $LOG
    $TADA decl_part_goto.a       >>& $LOG
    $TADA decl_part.a            >>& $LOG
    $TADA pre_parser.a           >>& $LOG
    $TADA compilation_unit_shift_reduce.a >>& $LOG
    $TADA compilation_unit_goto.a   >>& $LOG
    $TADA compilation_unit.a        >>& $LOG
    $TADA seq_of_stmts_shift_reduce.a >>& $LOG
    $TADA seq_of_stmts_goto.a       >>& $LOG
    $TADA seq_of_stmts.a            >>& $LOG
    $TADA calendar.a                >>& $LOG
    $TADA main.a                    >>& $LOG
    $TADA rwace.a                   >>& $LOG
    $TADA astd.a                    >>& $LOG
    $TADA expand.a                  >>& $LOG
    $TADA aprgut.a                  >>& $LOG
    $TADA alloc.a                   >>& $LOG
    $TADA cinfix.a                  >>& $LOG
    $TADA cdot.a                    >>& $LOG
    $TADA carray.a                  >>& $LOG

if (( $Caisop == "y" ) && ( $Xop == "y" )) then
    $TADA ebuilt.xcais.a            >>& $LOG
else if ( $Xop == "y" ) then
    $TADA ebuilt.x.a                >>& $LOG
else if ( $Caisop == "y" ) then
    $TADA ebuilt.cais.a             >>& $LOG
else
    $TADA ebuilt.a                  >>& $LOG
endif

    $TADA einfix.a                  >>& $LOG
    $TADA euser.a                   >>& $LOG
    $TADA eprag.a                   >>& $LOG
    $TADA eblock.a                  >>& $LOG
    $TADA epkgb.a                   >>& $LOG
    $TADA dtyped.a                  >>& $LOG

if ( $Xop == "y" ) then
    $TADA onlyx.a                   >>& $LOG
else
    $TADA nowin.a                   >>& $LOG
endif

    $TADA eobject.a                 >>& $LOG

if ( $Xop == "y" ) then
```

```
    $TADA xadt.a                    >>& $LOG
endif

    $TADA smgms.a                   >>& $LOG
    $TADA ace_input.a               >>& $LOG

if ( $Xop == "y" ) then
    $TADA ace_x_windows.a           >>& $LOG
    $TADA ace_widget.a              >>& $LOG
    $TADA ace_hp_widgets.a          >>& $LOG
endif

#
echo "Finally, the C subroutines"
echo ""
#
cd $ACE_CLIB
    date                            >>& $LOG

    $CC cflush.c                    >>& $LOG

    $CC sys_env.c                   >>& $LOG

if ( $Xop == "y" ) then
    $CC xwc.c                       >>& $LOG
    $CC xws.c                       >>& $LOG
endif

#
echo "Linking the objects"
echo ""
#
cd $TARGET/src
    date                            >>& $LOG

if ( $Xop == "y" ) then
  $TALD -p '$LIBRARIES_WITH_X' -o ACE main >>& $LOG
else
  $TALD -p '$LIBRARIES_NO_X $LIB_ARG' -o ACE main    >>& $LOG
endif

    date                            >>& $LOG

#
echo "Placing executable in $TARGET/bin"
echo ""
#
    mv -f ACE $TARGET/bin           >>& $LOG

#
echo "Build complete"
echo ""
```